Session 606

Direct to Java Client

Andreas Wendker
Manager, WebObjects Rapid Development

Introduction

- Java Client is a technology for creating distributed 3-tier desktop applications
  - Client displays user interface
  - Server accesses database
  - Database stores data
  - HTTP standard communication protocol
- Platform independent, localizable
- Designed for Intranet, not Internet

Introduction

- Direct to Java Client makes development of distributed desktop applications really easy!
- It's like instant ~~coffee~~ application, just add ~~water~~ EOModel

Introduction

- Direct to Java Client is an infrastructure designed to help you develop Java Client applications quickly
  - A lot of predefined application functionality
  - Generates user interfaces or parts of them dynamically

Introduction

- Direct to Java Client is an infrastructure designed to help you develop Java Client applications quickly
  - A lot of predefined application functionality
  - Generates user interfaces or parts of them dynamically
  - Saves a lot of time and money
  - Ideal for prototyping purposes

Introduction

- Direct to Java Client is an infrastructure designed to help you develop Java Client applications quickly
  - A lot of predefined application functionality
  - Generates user interfaces or parts of them dynamically
  - Saves a lot of time and money
  - Ideal for prototyping purposes
- Designed for extensibility
- Offers many customization techniques

What You'll Learn

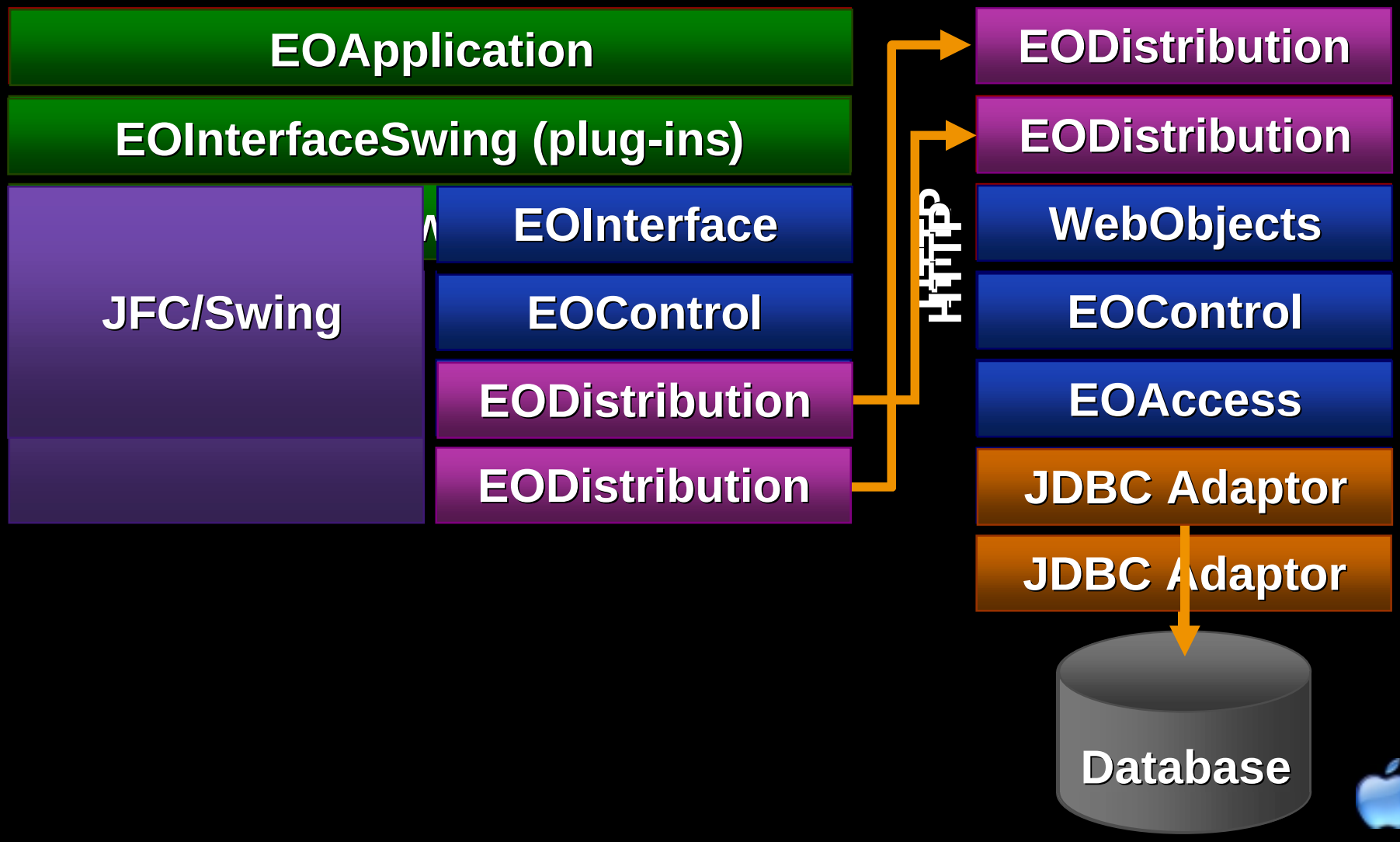
- How can you create Direct to Java Client applications?
- What do the default applications offer you?
- What architecture is used with Direct to Java Client applications?
- What are controllers?
- What is the Rule System?
- How can you customize applications?
- Why should you use Direct to Java Client?

Demo

Default Direct to Java Client Application

Ron Lue-Sang
WebObjects Engineer

Application Architecture



Controllers

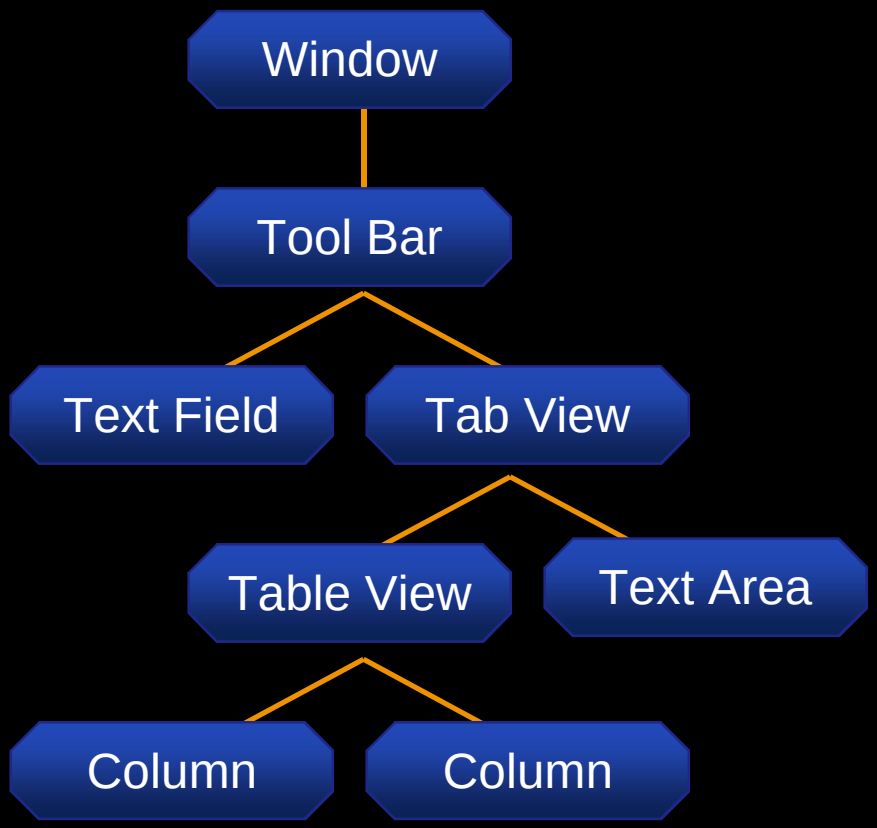
- Application functionality on the client is implemented as controllers
  - Controllers are the glue between the user interface objects (JFC/Swing) and business logic (EOF)
  - Controllers know how to create, lay out the necessary user interface and EOF objects like display groups, editing contexts

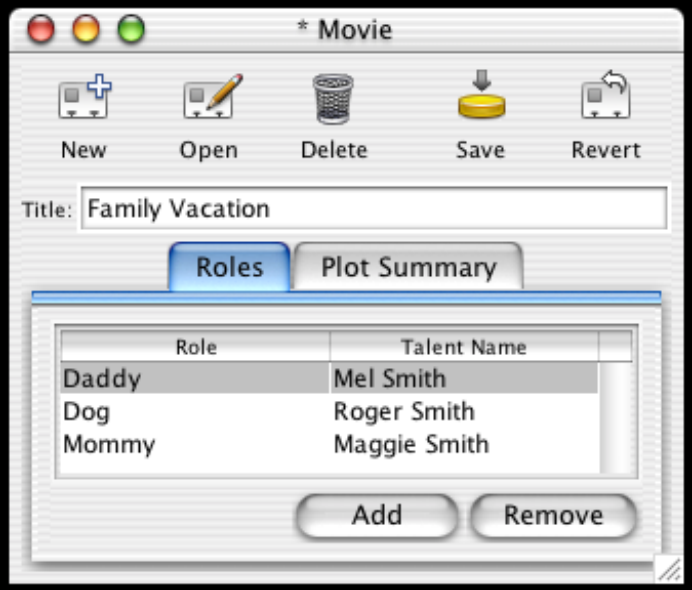
Controllers

- Types of controllers (all subclasses of EOController)
  - Application object
    (startup, configure menus, etc.)
  - User interface controllers
    (windows, tab views, various layout schemes)
  - Entity level controllers
    (business object management: query, list, form)
  - Property level controllers
    (text fields, combo boxes, pick lists, etc.)

Controller Hierarchy

- Controllers are organized in hierarchies





Controller Hierarchy in XML

- Controller hierarchies are defined in XML

```
<WINDOWCONTROLLER position="Center" label="Movie">
    <TOOLBARCONTROLLER>
        <TEXTFIELDCONTROLLER label="Title"/>
        <TABVIEWCONTROLLER>
            <TABLECONTROLLER label="Roles">
                <COLUMNCONTROLLER label="Role Name"/>
                <COLUMNCONTROLLER label="Talent Name"/>
            </TABLECONTROLLER>
            <TEXTAREACONTROLLER label="Plot Summary"/>
        </TABVIEWCONTROLLER>
    </TOOLBARCONTROLLER>
</WINDOWCONTROLLER>
```

Rule System

- Rule system generates information that describes how the client user interfaces look and behave
  - Located on the server side
  - Determines most aspects of the client user interfaces by analyzing EOModels
  - Most rules are abstract and can be applied to many applications (just define your corporate look and feel)
  - Convenient place for localization, etc.
  - Based on same technology as Direct to Web

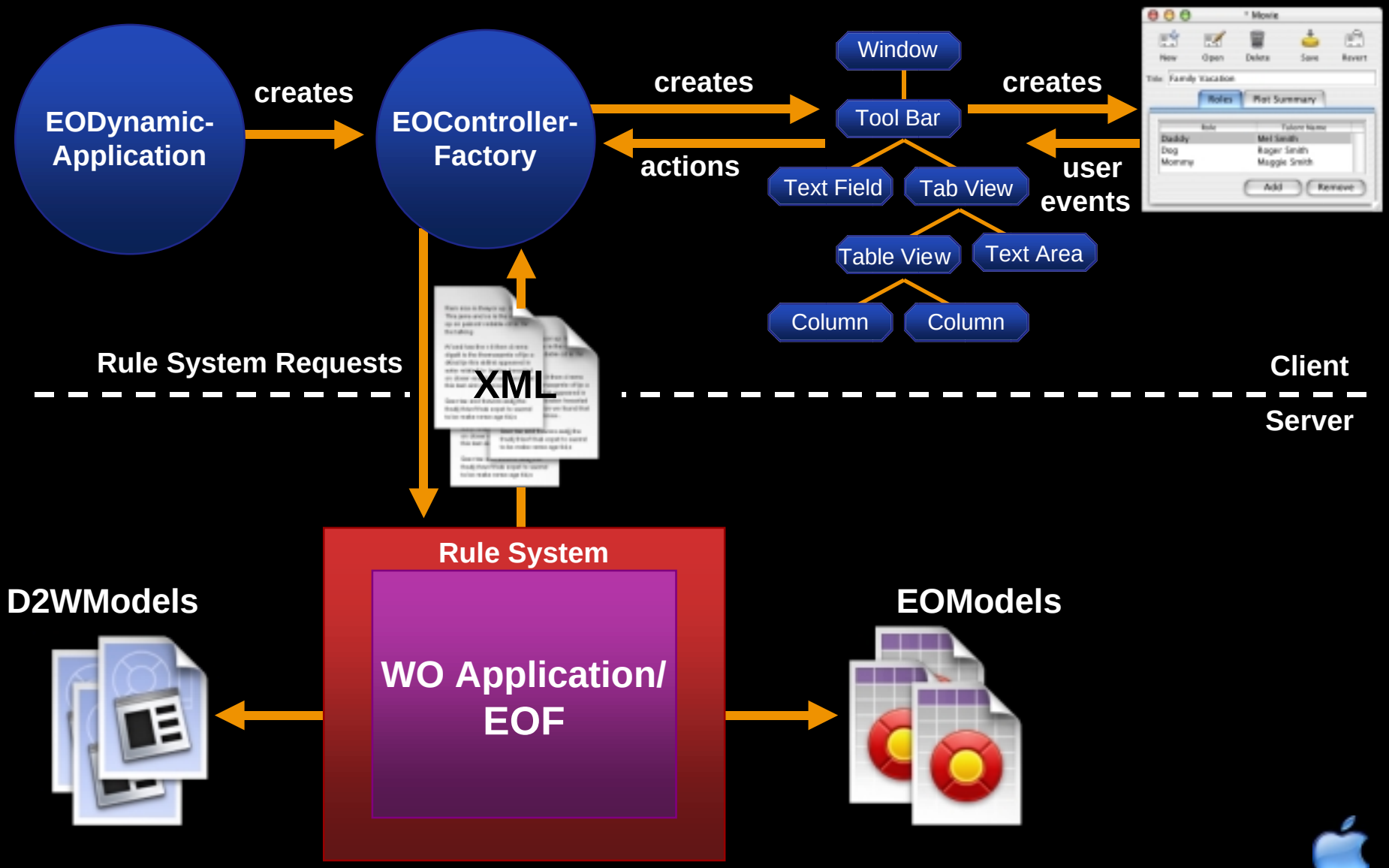
Rule System

- Rule system is a question/answer mechanism
  - What kind of windows are available?
  - Background colors for widgets?
  - Type of widgets (use text field or image view)?
- Direct to Java Client has a set of default rules
- It's easy to define custom rules

Rule System

- Clients ask only high-level questions, only need complete XML descriptions
- Clients ask for one window at a time
- Input to the rule system
  - Question (what kind of XML description is needed?)
  - Task (query, list, form)
  - Entity
  - Platform
  - Preferred languages

Data Flow

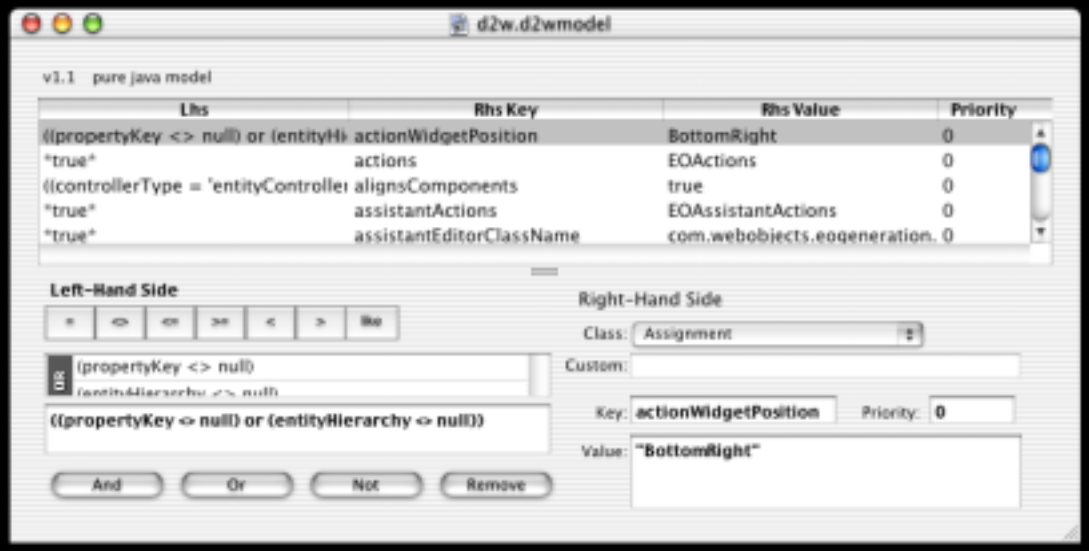

EODynamic-Application
creates
EOController-Factory
creates
actions
Window
Tool Bar
Text Field
Tab View
Table View
Text Area
Column
Column
creates
user events
Rule System Requests
XML
Client
Server
Rule System
WO Application/ EOF
D2WModels
EOModels

Rule System

- XML generation uses WebObjects component model
  - WOXMLNodes
  - WORepetitions, WOConditionals, etc.
  - D2WComponents connect to rule system
- D2WComponents use a D2WContext
  - D2WContext stores input parameters
  - D2WContext manages rules
  - D2WContext is entry point to fire rules

Development Tools

- Assistant: Basic tasks
- Rule Editor: Advanced customization
  - Example: Change start sequence, new types of windows (log-in panel)



Demo

Customization with Assistant, Rule Editor

Ron Lue-Sang
WebObjects Engineer

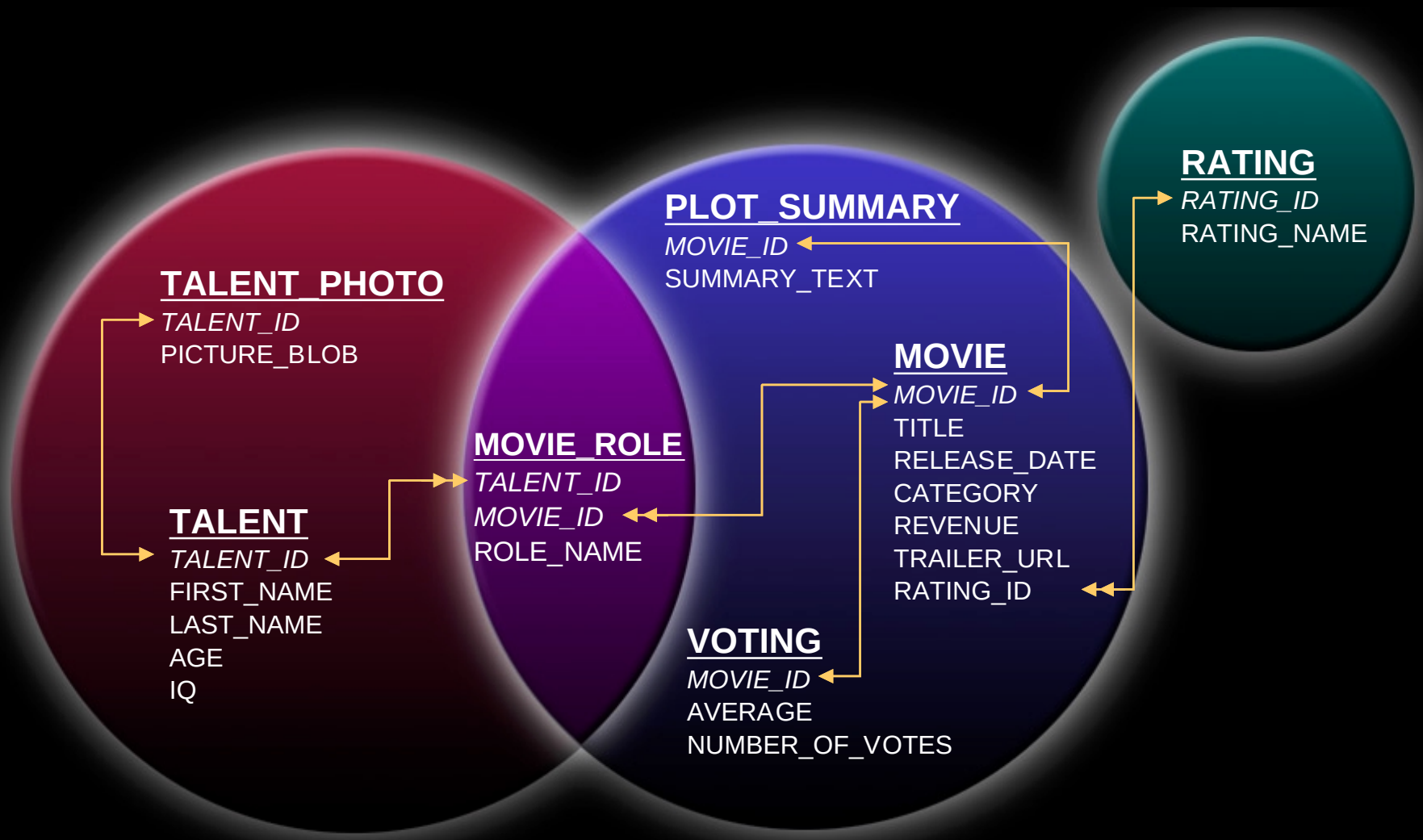
Rule System

- Example: Localization in rule system
  - Default rules leave creating labels to the client
  - Write rules to derive the label for a component based on information stored somewhere (for example in EOModel user info dictionaries)
    - Use multiple rules for different controller types
- Note: Localization/platform specific resource lookup provided by default

Default Rules

- Direct to Java Client's default rules mostly analyze EOModels
- Step 1: Categorize entities:
  - Main entities
  - Enumeration entities
  - "Dependent" entities

Default Rules: EOModel Analysis



Default Rules

- Step 2: Categorize properties
  - Query (entering qualifiers)
  - List (displaying tables)
  - Form (input fields)
  - Identify (referencing objects)

Default Rules

- Step 3: Determine available windows
  - Query Window
  - Enumeration Window
  - Form windows with detail user interfaces
  - Modal query dialogs

Default Rules

- Step 4: Determine widget layout
  - Rows and columns for attributes (depends on number of attributes)
  - Mandatory to-one relationships to main entities in boxes
  - Other relationships in switch controller (usually a tab view)
  - Simplified user interface for relationships to enumeration entities: combo boxes or pick lists

Customization Techniques

1. Assistant
   Rule Editor
2. Freezing XML
   Freezing Interface Files
3. Subclass controller classes
   Programmatically change factory behavior

Customization Techniques

- Freezing XML
  - Bypass rule system
  - Use static XML or mix with dynamic elements
  - Complete windows or only parts of them
  - Use your own templates
  - Need rule to “register” XML component

Customization Techniques

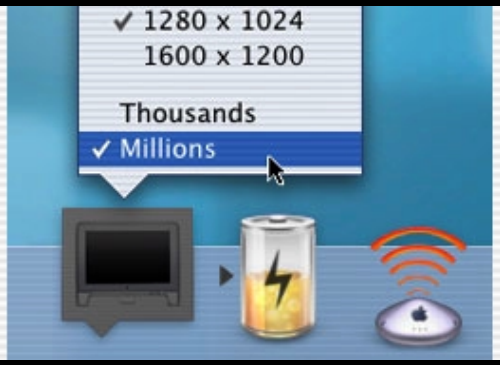
- Freezing interface files
  - Full control of widget layout
  - Complete windows or only parts of them
  - Make use of “subcontroller areas”
- Need rule to “register” interface files
- Need rule to suppress standard subcontrollers

Customization Techniques

- Subclass controller classes
  - Support for custom widgets/associations
  - Different layout mechanisms
  - Special qualifiers/fetch specifications
  - Additional actions
- Programmatically change factory behavior
  - Completely customize flow of user interface

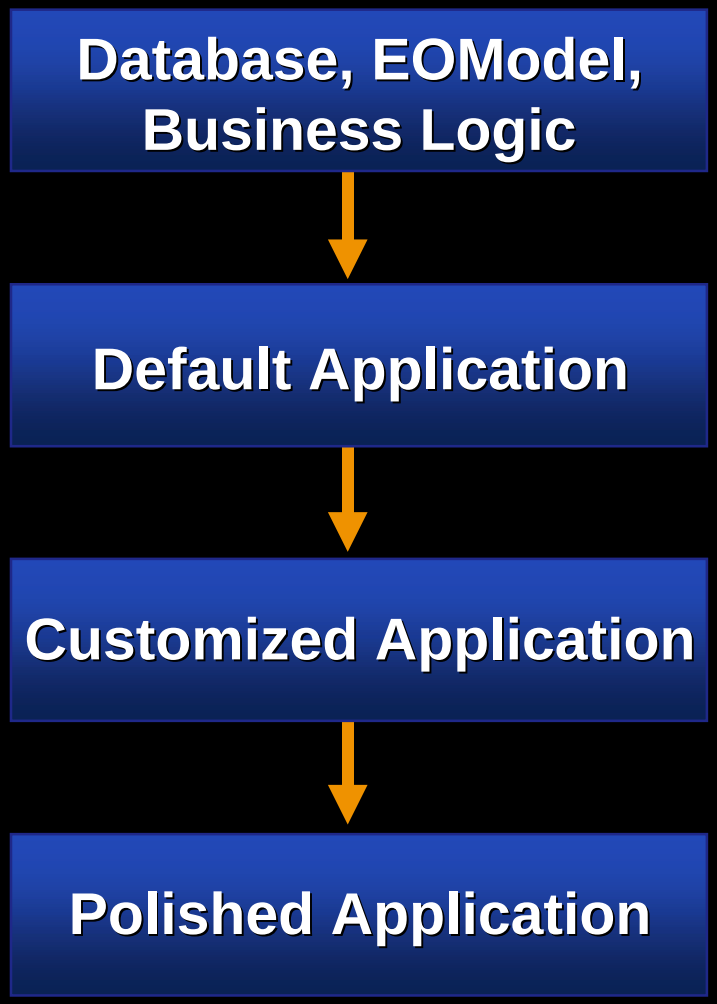
Demo

Advanced Customization



Ron Lue-Sang
WebObjects Engineer

Development Process Changes



- Creating database and EOModel sufficient to run default application
- Customization is easy and quick (efficient prototyping)
- Delay polish (user interface layout, application logic) until business logic finalized

Benefits

- Fast, efficient development
- Quick reaction to requirement/database changes
- Reduced testing time
- Maintenance much easier
- Focus on business logic

Tips for Using Direct to Java Client

- Try using it for every project
  - Direct to Java Client is Java Client, it's not different
  - Prototyping very easy
  - Smooth transition into "traditional" Java Client
- Define reusable frameworks for whole application suite
  - Frameworks with common rules
  - Frameworks for log-in, reporting, etc.

Tips for Using Direct to Java Client

- Localization especially easy
- Deployment relatively simple
  - Less custom classes required in comparison to “standard” Java Client applications
  - Many changes/bug fixes can be applied on the server side, don’t require client upgrade

Summary

- Direct to Java Client makes your life easy
  - Dynamic generation of applications simply based on EOModel
  - Controller classes provide predefined, tested functionality
  - Powerful customization
  - Smooth transition into traditional development
  - Designed for extensibility

Guest

Bill Baumgarner
Partner—Chief Technology Officer
CodeFab, Inc.

WebObjects Lab

- Located downstairs in Room K
- Lab hours
  - Monday 12:00pm–6:00pm
  - Tuesday 10:30am–6:00pm
  - Wednesday 10:30am–6:00pm
  - Thursday 10:30am–6:00pm
  - Friday 10:30am–3:30pm

☑ Vote for WebObjects as Best App Server

http://developer.apple.com/webobjects

Roadmap

612 Direct to Web	Room A1 **Thurs., 2:00pm**
613 Advanced Enterprise Objects Framework	Room A1 **Thurs., 3:30pm**
614 Advanced Java Client	Room A1 **Thurs., 5:00pm**

Who to Contact

Toni Trujillo Vian
Director, WebObjects Engineering
webobjects@apple.com

Bob Fraser
WebObjects Product Manager
webobjects@apple.com

iServices Consulting, Integration, Training & Certification
(800) 848-6398
iservices@apple.com

For more information:
www.apple.com/webobjects
developer.apple.com/webobjects
www.apple.com/iservices
www.info.apple.com/webobjects

Subscribe to:
webobjects-announce@apple.com

Q&A

Toni Trujillo Vian
Director, WebObjects Engineering
webobjects@apple.com

http://developer.apple.com/events/urls.html

WWDC

Worldwide Developers Conference 2001